C 8

```
   OOOOOOOOO     PPPPPPPPPPPP        CCCCCCCCCCCC     OOOOOOOOO     MMM         MMM
   OOOOOOOOO     PPPPPPPPPPPP        CCCCCCCCCCCC     OOOOOOOOO     MMM         MMM
   OOOOOOOOO     PPPPPPPPPPPP        CCCCCCCCCCCC     OOOOOOOOO     MMM         MMM
OOO         OOO  PPP         PPP  CCC              OOO         OOO  MMMMMM   MMMMMM
OOO         OOO  PPP         PPP  CCC              OOO         OOO  MMMMMM   MMMMMM
OOO         OOO  PPP         PPP  CCC              OOO         OOO  MMMMMM   MMMMMM
OOO         OOO  PPP         PPP  CCC              OOO         OOO  MMM   MMM   MMM
OOO         OOO  PPP         PPP  CCC              OOO         OOO  MMM   MMM   MMM
OOO         OOO  PPP         PPP  CCC              OOO         OOO  MMM   MMM   MMM
OOO         OOO  PPPPPPPPPPPP     CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPPPPPPPPPPP     CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPPPPPPPPPPP     CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPP              CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPP              CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPP              CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPP              CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPP              CCC              OOO         OOO  MMM         MMM
OOO         OOO  PPP              CCC              OOO         OOO  MMM         MMM
   OOOOOOOOO     PPP                 CCCCCCCCCCCC     OOOOOOOOO     MMM         MMM
   OOOOOOOOO     PPP                 CCCCCCCCCCCC     OOOOOOOOO     MMM         MMM
   OOOOOOOOO     PPP                 CCCCCCCCCCCC     OOOOOOOOO     MMM         MMM
```

```
_$2

Sym
---
ALL

ASC

BOD
BOD
BOD
BOD
BOD
BOD
BOD
BOD
BUG
BYP
CAN
CAN
CAN
CHE
CHE

CLU
CLU
CLU
CLU
CLU
CLU
CLU

CLU
CLU
CLU
CLU
CLU
CLU
CLU

CLU
CLU
```

```
**FILE**ID**TIMESTAMP                                        N 9

TTTTTTTTTT   IIIIII   MM      MM EEEEEEEEEE   SSSSSSSS TTTTTTTTTT   AAAAAA   MM      MM PPPPPPPP
TTTTTTTTTT   IIIIII   MM      MM EEEEEEEEEE   SSSSSSSS TTTTTTTTTT   AAAAAA   MM      MM PPPPPPPP
    TT         II     MMMM  MMMM EE         SS             TT     AA      AA MMMM  MMMM PP      PP
    TT         II     MMMM  MMMM EE         SS             TT     AA      AA MMMM  MMMM PP      PP
    TT         II     MM  MM  MM EE         SS             TT     AA      AA MM  MM  MM PP      PP
    TT         II     MM  MM  MM EE         SS             TT     AA      AA MM  MM  MM PP      PP
    TT         II     MM      MM EEEEEEEE     SSSSSS       TT     AA      AA MM      MM PPPPPPPP
    TT         II     MM      MM EEEEEEEE     SSSSSS       TT     AA      AA MM      MM PPPPPPPP
    TT         II     MM      MM EE                 SS     TT     AAAAAAAAAA MM      MM PP
    TT         II     MM      MM EE                 SS     TT     AAAAAAAAAA MM      MM PP
    TT         II     MM      MM EE                 SS     TT     AA      AA MM      MM PP                  ....
    TT         II     MM      MM EE                 SS     TT     AA      AA MM      MM PP                  ....
    TT       IIIIII   MM      MM EEEEEEEEEE SSSSSSSS       TT     AA      AA MM      MM PP                  ....
    TT       IIIIII   MM      MM EEEEEEEEEE SSSSSSSS       TT     AA      AA MM      MM PP                  ....

LL           IIIIII     SSSSSSSS
LL           IIIIII     SSSSSSSS
LL             II     SS
LL             II     SS
LL             II     SS
LL             II     SS
LL             II       SSSSSS
LL             II       SSSSSS
LL             II             SS
LL             II             SS
LL             II             SS
LL             II             SS
LLLLLLLLLL   IIIIII   SSSSSSSS
LLLLLLLLLL   IIIIII   SSSSSSSS
```

```
MODULE  OPC$TIMESTAMP   (
                        LANGUAGE (BLISS32),
                        IDENT = 'V04-000'
                        ) =

!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************

!++
! FACILITY:
!
!       OPCOM
!
! ABSTRACT:
!
!       This module contains all the various and sundry general
!       purpose utility routines used by OPCOM's request handlers.
!
! Environment:
!
!       VAX/VMS operating system.
!
! Author:
!
!       Steven T. Jeffreys
!
! Creation date:
!
!       March 10, 1981
!
! Revision history:
!
!       V03-005 CWH3005         CW Hobbs                        25-Jul-1984
!               Tune the workset purge algorithm to eliminate purges on
!               a quiet OPCOM.
!
!       V03-004 CWH3169         CW Hobbs                        5-May-1984
```

```
0 !                     Second pass for cluster-wide OPCOM:
0 !                      - Slow from 15 second timestamps to 5 minute timestamps.
0 !                      - No longer do configures during a timestamp.
0 !                      - Purge the working set on an hourly basis.
0 !
0 !             V03-003 CWH3003         CW Hobbs                        16-Sep-1983
0 !                     Clear ioerr flag at each timestamp
0 !
0 !             V03-002 CWH3002         CW Hobbs                        30-Jul-1983
0 !                     Various and sundry things to make OPCOM distributed
0 !                     across the cluster.
0 !
0 !             V03-001 STJ3031         Steven T. Jeffreys,             05-Oct-1982
0 !                     - Added the IMPLICITLY_CANCELED routine.
0 !                     - Added the IMPLIED_CANCEL routine.
0 !                     - Added the IMPLIED_DISABLE dummy routine.
0 !                     - Flush the logfile if it has been written to.
0 !
0 !--
0
1 BEGIN                                                 ! Start of TIMESTAMP
1
1 LIBRARY 'SYS$LIBRARY:LIB.L32';
1 LIBRARY 'LIB$:OPCOMLIB';
1
1 FORWARD ROUTINE
1               TIME_STAMP        : NOVALUE,    ! Periodic wakeup routine
1               IMPLICITLY_CANCELED,            ! Determine if request canceled
1               IMPLIED_CANCEL    : NOVALUE,    ! Perform implicit request cancellation
1               IMPLIED_DISABLE : NOVALUE;      ! Perform implicit operator disable
1
1 BUILTIN
1               INSQUE,                         ! Insert entry onto a queue
1               REMQUE;                         ! Remove entry from a queue
1
```

```
1 GLOBAL ROUTINE TIME_STAMP : NOVALUE =
1
1 !++
1 ! Functional description:
1 !
1 !       TIME_STAMP is an AST service routine that is executed periodically
1 !       to cause OPCOM to perform its periodic timestamp function and then
1 !       issue another timer AST request.  The timestamp function is to remind
1 !       all operators of outstanding requests.  If the operator has the
1 !       NOREMIND option set, then the operator will not be reminded.
1 !       TIME_STAMP uses an interlock mechanism to insure that the timestamp
1 !       will not occur at an inappropriate time for OPCOM.
1 !
1 !       No timestamp message is explicitly logged, but messages may be logged
1 !       as operators are implicitly disabled and requests are canceled.
1 !
1 ! Input:
1 !
1 !       None.
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       None.
1 !
1 ! Side effects:
1 !
1 !       None.
1 !
1 ! Routine value:
1 !
1 !       None.
1 !--
1
2 BEGIN                                               ! Start of TIME_STAMP
2
2 EXTERNAL ROUTINE
2       ALLOCATE_DS,                                  ! Get structure
2       CLUSMSG_ACK_PLEASE,                           ! Request acknowledgement from a remote node
2       CLUSMSG_STATE_SEND,                           ! Tell cluster about current operators and requests
2       DEALLOCATE_RQCB,                              ! Return RQCB structure
2       FORMAT_MESSAGE,                               ! Format a message
2       LOG_MESSAGE,                                  ! Send a message to the log file
2       NOTIFY_LISTED_OPERATORS;                      ! Notify a given operator
2
2 EXTERNAL LITERAL
2       RQCB_K_TYPE,                                  ! Type code for RQCB structure
2       MIN_SCOPE,                                    ! Minimum scope value
2       MAX_SCOPE;                                    ! Maximum scope value
2
```

```
2 EXTERNAL
2         LOGFILE_RAB     : $bblock,              ! RAB for operator logfile
2         OCD_VECTOR      : VECTOR,               ! OCD list heads
2         GLOBAL_STATUS   : BITVECTOR,            ! Global status bits for OPCOM
2         NOD_HEAD        : VECTOR [2, LONG],     ! Head of node queue
2         WAIT_DELTA      : $ref_bblock,          ! Delta time quadword
2         SYI_SWPOUTPGCNT : LONG,                 ! Swap out page count
2         LOGTIME_COUNTER : LONG;                 ! Counter for log file timestamp messages
2
2 GLOBAL                                          ! Make it easy to find with SDA
2         PURGE_LIMIT     : LONG;                 ! Purge work set if above this value
2
2 OWN
2         GPGCNT          : LONG,                 ! Global pages in working set
2         PPGCNT          : LONG,                 ! Process pages in working set
2         JPI_WSITEMS     : VECTOR [8, LONG]      ! Item list to get working set items
2                         PRESET ([0] = (jpi$_gpgcnt^16 OR 4),
2                                 [1] = GPGCNT,
2                                 [2] = 0,
2                                 [3] = (jpi$_ppgcnt^16 OR 4),
2                                 [4] = PPGCNT,
2                                 [5] = 0,
2                                 [6] = 0,            ! End of item list, head of $PURGWS addr desc
2                                 [7] = %X'7FFFFFFF');    ! End of $PURGWS addr desc
2
2 LOCAL
2         RQST            : $ref_bblock,          ! RQCB (request) data structure
2         NEXT_RQST       : $ref_bblock,          ! ditto
2         RQST_COUNT      : LONG,                 ! Count of requests in list
2         NOD             : $ref_bblock,          ! Node data structure
2         OCD             : $ref_bblock,          ! OCD data structure
2         NEXT_OCD        : $ref_bblock,          ! ditto
2         OCD_COUNT       : LONG,                 ! Count of OCDs in list
2         STATUS          : LONG;
2
2 !
2 ! If shutdown is pending, then do nothing.
2 !
2 IF .GLOBAL_STATUS [GBLSTS_K_SHUTDOWN_PENDING]
2 THEN
3     BEGIN
3     GLOBAL_STATUS [GBLSTS_K_TIMESTAMP_PENDING] = FALSE;
3     RETURN;
2     END;
2 !
2 ! Set GBLSTS_K_TIMESTAMP_PENDING.  If OPCOM is busy, then return.
2 ! If not, then set GBLSTS_K_BUSY to prevent another timestamp AST from arriving.
2 !
2 GLOBAL_STATUS [GBLSTS_K_TIMESTAMP_PENDING] = TRUE;
2 IF .GLOBAL_STATUS [GBLSTS_K_BUSY]
2 THEN
2     RETURN;
2 GLOBAL_STATUS [GBLSTS_K_BUSY] = TRUE;
2
2 !
2 ! Every twelve timestamps (once an hour), stamp the log file.  Also, since we might
2 ! have a lot of garbage sitting in memory, flush the working set so that we do not
```

```
2 ! eat up unnecessary pages on small systems.
2 !
2 IF (LOGTIME_COUNTER = .LOGTIME_COUNTER + 1) GEQ 12
2 THEN
3     BEGIN
3     !+
3     ! Start of 60 minute timestamp
3     !-
3     LOCAL
3         MSGVEC : VECTOR [2, LONG],       ! Temporary vector for message
3         LOG_RQCB : $ref_bblock;
3     LOGTIME_COUNTER = 0;
3     IF ALLOCATE_DS (RQCB_K_TYPE, LOG_RQCB)
3     THEN
4         BEGIN
4         MSGVEC [0] = OPC$_LOGTIME;
4         MSGVEC [1] = 0;
4         FORMAT_MESSAGE (.LOG_RQCB, MSGVEC);
4         LOG_MESSAGE (.LOG_RQCB);
4         DEALLOCATE_RQCB (.LOG_RQCB);
3         END;
3     !
3     ! Flush the working set, but first check to make sure that we are big enough to need it
3     ! Note also that by flushing before the 5 minute section, we will most likely fault in
3     ! the code and data needed by the timestamp from the lists, rather than doing real I/O.
3     !
4     IF NOT (STATUS = $GETJPI (ITMLST=JPI_WSITEMS))
3     THEN
3         $signal_stop (.STATUS);
3     IF .PPGCNT+.GPGCNT GTR .PURGE_LIMIT
3     THEN
4         BEGIN
4         PURGE_LIMIT = 0;                            ! Reset so we will recalculate what we need
4         $PURGWS (INADR=JPI_WSITEMS[6]);            ! Reuse a longword of the item list
3         END;
3     !+
3     ! End of 60 minute
3     !-
2     END;
2
2 !+
2 ! Start of 5 minute timestamp
2 !-
2
2 !
2 ! For each request outstanding, notify all interested operators.
2 !
2 ! Before notifying the interested operators, check to see if the request
2 ! has been implicitly canceled.  If so, insert it on a special queue for
2 ! processing later in this routine.
2 !
2 ! Also note that as this is happening, implicitly disabled operators are
2 ! being processed.  They too will be removed from the data base later in
2 ! this routine.
2 !
2 INCR I FROM MIN_SCOPE TO MAX_SCOPE DO
```

```
3          BEGIN
3          !
3          ! For each each class of operator (SYSTEM, GROUP, USER) ...
3          !
3          NEXT_OCD = .OCD_VECTOR [(.I-1)*2];              ! Get first OCD in list
3          INCR J FROM 1 TO .OCD_VECTOR [(.I-1)*2+1] DO
4              BEGIN
4              !
4              ! For each OCD in the operator class list...
4              !
4              OCD = .NEXT_OCD;                                ! Get current OCD address
4              NEXT_OCD = .OCD [OCD_L_FLINK];                  ! Get next OCD address
4              NEXT_RQST = .OCD [OCD_L_RQSTFLINK];             ! Get first request address
4              INCR K FROM 1 TO .OCD [OCD_W_RQSTCOUNT] DO
5                  BEGIN
5                  !
5                  ! For each request in the OCD list...
5                  !
5                  RQST = .NEXT_RQST;                              ! Get current request address
5                  NEXT_RQST = .RQST [RQCB_L_FLINK];               ! Get next request address
5                  IF NOT IMPLICITLY_CANCELED (.RQST)
5                  THEN
5                      !
5                      ! The reply mailbox exists.  Inform operators of the request.
5                      !
5                      NOTIFY_LISTED_OPERATORS (.RQST)
4                  END;
3              END;
2          END;
2      !
2      ! After sweeping through the data base, we may have discovered some
2      ! implicitly canceled requests and implicitly disabled operators.
2      ! Process them now.  The requests should be done first, as yet more
2      ! implicitly disabled operators may turn up.
2      !
2      IMPLIED_CANCEL ();
2      IMPLIED_DISABLE ();
2      !
2      ! Make a scan through the node database
2      !
2      NOD = .NOD_HEAD [0];
2      WHILE .NOD NEQ NOD_HEAD [0]
2      DO
3          BEGIN
3          !
3          ! Clear the error message flag.  This limits the rate of OPC$_CLUSCOMM error messages to
3          ! one per five minutes.
3          !
3          NOD [NOD_V_IOERR_DISPLAYED] = FALSE;
3          !
3          ! If we have any nodes in "START" state, then request an acknowledgement from them.
3          !
3          IF .NOD [NOD_B_STATE] EQL NOD_K_STATE_START
3          THEN
4              BEGIN
4              NOD [NOD_V_ACK_PEND] = FALSE;           ! Clear so that we can
4              CLUSMSG_ACK_PLEASE (.NOD);              !  request an acknowledgement
```

```
3         END;
3     NOD = .NOD [NOD_L_FLINK];
2     END;
2 !
2 ! If the operator logfile was written to since the last timestamp operation,
2 ! flush the contents of the RMS buffers to the disk.  This also has the effect
2 ! of writing the file header, so the information is not lost in the event of
2 ! a system crash.  This is necessary because the log file is kept open until
2 ! explicitly closed via REPLY/[NO]LOG.
2 !
2 !
2 IF .GLOBAL_STATUS [GBLSTS_K_FLUSH_PENDING]
2 THEN
3     BEGIN
3     GLOBAL_STATUS [GBLSTS_K_FLUSH_PENDING] = FALSE;
3     $FLUSH (RAB = LOGFILE_RAB);
2     END;
2 !
2 ! If we purged the working set on this pass, then save the size we have now.
2 ! This lets us react to peaks in working set use, without a lot of faults
2 ! during periods of non-activity.
2 !
2 IF .PURGE_LIMIT EQL 0
2 THEN
3     BEGIN
3     REGISTER
3         SWAPO, LIMIT;
4     IF NOT (STATUS = $GETJPI (ITMLST=JPI_WSITEMS))
3     THEN
3         $signal_stop (.STATUS);
3     !
3     ! Set new value to 10 more pages than we are currently using, but no lower
3     ! than swap-out-page-count and no higher than 3 times swap-out-page-count.
3     !
3     SWAPO = .SYI_SWPOUTPGCNT;                   ! Get it into a register
3     LIMIT = MAX (.PPGCNT+.GPGCNT+10, .SWAPO);   ! Limit is larger of swapo and 10 more than current
3     SWAPO = 3 * .SWAPO;                         ! Compute the max
3     PURGE_LIMIT = MIN (.SWAPO, .LIMIT);         ! Actual limit is smaller of the two
2     END;
2 !
2 ! Queue another timer ast.
2 !
2 GLOBAL_STATUS [GBLSTS_K_BUSY] = FALSE;
2 GLOBAL_STATUS [GBLSTS_K_TIMESTAMP_PENDING] = FALSE;
3 IF NOT (STATUS = $SETIMR (EFN = EFN_K_TIME_STAMP, DAYTIM = WAIT_DELTA, ASTADR = TIME_STAMP))
2 THEN
2     $signal_stop (.STATUS);
2
1 END;                                            ! End of TIME_STAMP

                                                  .TITLE  OPC$TIMESTAMP
                                                  .IDENT  \V04-000\

                                                  .PSECT  $OWN$,NOEXE,2

                                  00000 GPGCNT:   .BLKB   4
```

```
                                      00004 PPGCNT:  .BLKB    4
                             030C0004  00008 JPI_WSITEMS:
                                                     .LONG    51118084                                    :
                             00000000' 0000C         .ADDRESS GPGCNT                                      :
                   030D0004  00000000  00010         .LONG    0, 51183620                                 :
                             00000000' 00018         .ADDRESS PPGCNT                                      :
         7FFFFFFF  00000000  00000000  0001C         .LONG    0, 0, 2147483647                            :

                                                     .PSECT   $GLOBAL$,NOEXE,2

                                      00000 PURGE_LIMIT::
                                                     .BLKB    4

                                                     .EXTRN   ALLOCATE_DS, CLUSMSG_ACK_PLEASE
                                                     .EXTRN   CLUSMSG_STATE_SEND
                                                     .EXTRN   DEALLOCATE_RQCB
                                                     .EXTRN   FORMAT_MESSAGE, LOG_MESSAGE
                                                     .EXTRN   NOTIFY_LISTED_OPERATORS
                                                     .EXTRN   RQCB_K_TYPE, MIN_SCOPE
                                                     .EXTRN   MAX_SCOPE, LOGFILE_RAB
                                                     .EXTRN   OCD_VECTOR, GLOBAL_STATUS
                                                     .EXTRN   NOD_HEAD, WAIT_DELTA
                                                     .EXTRN   SYI_SWPOUTPGCNT
                                                     .EXTRN   LOGTIME_COUNTER
                                                     .EXTRN   SYS$GETJPI, LIB$STOP
                                                     .EXTRN   SYS$PURGWS, SYS$FLUSH
                                                     .EXTRN   SYS$SETIMR

                                                     .PSECT   $CODE$,NOWRT,2

                                 0FFC 00000          .ENTRY   TIME_STAMP, Save R2,R3,R4,R5,R6,R7,R8,R9,-  ; 0093
                                                              R10,R11                                     :
                  5E          0C   C2 00002          SUBL2    #12, SP                                     :
                  06    0000G CF   E9 00005          BLBC     GLOBAL_STATUS, 1$                           ; 0188
          0000G   CF          20   8A 0000A          BICB2    #32, GLOBAL_STATUS                          ; 0191
                                   04 0000F          RET                                                  ; 0190
          0000G   CF          20   88 00010 1$:      BISB2    #32, GLOBAL_STATUS                          ; 0198
    01    0000G   CF          06   E1 00015          BBC      #6, GLOBAL_STATUS, 2$                       ; 0199
                                   04 0001B          RET                                                  :
          0000G   CF      40  8F   88 0001C 2$:      BISB2    #64, GLOBAL_STATUS                          ; 0202
    50    0000G   CF          01   C1 00022          ADDL3    #1, LOGTIME_COUNTER, R0                     ; 0209
          0000G   CF          50   D0 00028          MOVL     R0, LOGTIME_COUNTER                         :
                  0C          50   D1 0002D          CMPL     R0, #12                                     :
                              72   19 00030          BLSS     5$                                          :
                        0000G CF   D4 00032          CLRL     LOGTIME_COUNTER                             ; 0218
                              5E   DD 00036          PUSHL    SP                                          ; 0219
                    00000000G 8F   DD 00038          PUSHL    #RQCB_K_TYPE                                :
          0000G   CF          02   FB 0003E          CALLS    #2, ALLOCATE_DS                             :
                  24          50   E9 00043          BLBC     R0, 3$                                      :
             04   AE 00058099 8F   D0 00046          MOVL     #360601, MSGVEC                             ; 0222
                          08  AE   D4 0004E          CLRL     MSGVEC+4                                    ; 0223
                          04  AE   9F 00051          PUSHAB   MSGVEC                                      ; 0224
                          04  AE   DD 00054          PUSHL    LOG_RQCB                                    :
          0000G   CF          02   FB 00057          CALLS    #2, FORMAT_MESSAGE                          :
                              6E   DD 0005C          PUSHL    LOG_RQCB                                    ; 0225
          0000G   CF          01   FB 0005E          CALLS    #1, LOG_MESSAGE                             :
                              6E   DD 00063          PUSHL    LOG_RQCB                                    ; 0226
                   0000G  CF                 01  FB 00065           CALLS    #1, DEALLOCATE_RQCB                     ;
                                             7E  7C 0006A 3$:       CLRQ     -(SP)                                   ; 0233
                                             7E  D4 0006C           CLRL     -(SP)                                   ;
                                  0000'      CF  9F 0006E           PUSHAB   JPI_WSITEMS                             ;
                                             7E  7C 00072           CLRQ     -(SP)                                   ;
                                             7E  D4 00074           CLRL     -(SP)                                   ;
               00000000G  00                 07  FB 00076           CALLS    #7, SYS$GETJPI                          ;
                          5B                 50  D0 0007D           MOVL     R0, STATUS                              ;
                          03                 5B  E8 00080           BLBS     STATUS, 4$                              ;
                                           0125  31 00083           BRW      19$                                     ;
        50         0000'  CF      0000'      CF  C1 00086 4$:       ADDL3    GPGCNT, PPGCNT, R0                      ; 0236
                   0000'  CF                 50  D1 0008E           CMPL     R0, PURGE_LIMIT                         ;
                                             0F  15 00093           BLEQ     5$                                      ;
                                  0000'      CF  D4 00095           CLRL     PURGE_LIMIT                             ; 0239
                                  0000'      CF  9F 00099           PUSHAB   JPI_WSITEMS+24                          ; 0240
               00000000G  00                 01  FB 0009D           CALLS    #1, SYS$PURGWS                          ;
        52     00000000G  8F                 01  C3 000A4 5$:       SUBL3    #1, #MIN_SCOPE, I                       ; 0263
                                             45  11 000AC           BRB      11$                                     ;
        50                52                 01  78 000AE 6$:       ASHL     #1, I, R0                               ; 0268
                          58      0000GCF40      D0 000B2           MOVL     OCD_VECTOR-8[R0], NEXT_OCD              ;
                          5A      0000GCF40      D0 000B8           MOVL     OCD_VECTOR-4[R0], R10                   ; 0269
                                             55  D4 000BE           CLRL     J                                       ;
                                             2D  11 000C0           BRB      10$                                     ;
                          53                 58  D0 000C2 7$:       MOVL     NEXT_OCD, OCD                           ; 0274
                          58                 63  D0 000C5           MOVL     (OCD), NEXT_OCD                         ; 0275
                          59         3C      A3  D0 000C8           MOVL     60(OCD), NEXT_RQST                      ; 0276
                          57         3A      A3  3C 000CC           MOVZWL   58(OCD), R7                             ; 0277
                                             54  D4 000D0           CLRL     K                                       ;
                                             17  11 000D2           BRB      9$                                      ;
                          56                 59  D0 000D4 8$:       MOVL     NEXT_RQST, RQST                         ; 0282
                          59                 66  D0 000D7           MOVL     (RQST), NEXT_RQST                       ; 0283
                                             56  DD 000DA           PUSHL    RQST                                    ; 0284
                   0000V  CF                 01  FB 000DC           CALLS    #1, IMPLICITLY_CANCELED                 ;
                          07                 50  E8 000E1           BLBS     R0, 9$                                  ;
                                             56  DD 000E4           PUSHL    RQST                                    ; 0289
                   0000G  CF                 01  FB 000E6           CALLS    #1, NOTIFY_LISTED_OPERATORS             ;
        E5                54                 57  F3 000EB 9$:       AOBLEQ   R7, K, 8$                               ; 0284
        CF                55                 5A  F3 000EF 10$:      AOBLEQ   R10, J, 7$                              ; 0269
        B3                52      00000000G  8F  F3 000F3 11$:      AOBLEQ   #MAX_SCOPE, I, 6$                       ; 0263
                   0000V  CF                 00  FB 000FB           CALLS    #0, IMPLIED_CANCEL                      ; 0299
                   0000V  CF                 00  FB 00100           CALLS    #0, IMPLIED_DISABLE                     ; 0300
                          53      0000G      CF  D0 00105           MOVL     NOD_HEAD, NOD                           ; 0304
                          50      0000G      CF  9E 0010A 12$:      MOVAB    NOD_HEAD, R0                            ; 0305
                          50                 53  D1 0010F           CMPL     NOD, R0                                 ;
                                             1A  13 00112           BEQL     14$                                     ;
                      2A  A3                 04  8A 00114           BICB2    #4, 42(NOD)                             ; 0312
                          02         22      A3  91 00118           CMPB     34(NOD), #2                             ; 0316
                                             0B  12 0011C           BNEQ     13$                                     ;
                      2A  A3                 01  8A 0011E           BICB2    #1, 42(NOD)                             ; 0319
                                             53  DD 00122           PUSHL    NOD                                     ; 0320
                   0000G  CF                 01  FB 00124           CALLS    #1, CLUSMSG_ACK_PLEASE                  ;
                          53                 63  D0 00129 13$:      MOVL     (NOD), NOD                              ; 0322
                                             DC  11 0012C           BRB      12$                                     ; 0305
                                  0000G      CF  95 0012E 14$:      TSTB     GLOBAL_STATUS                           ; 0332
                                             11  18 00132           BGEQ     15$                                     ;
                   0000G  CF         80      8F  8A 00134           BICB2    #128, GLOBAL_STATUS                     ; 0335
                                  0000G      CF  9F 0013A           PUSHAB   LOGFILE_RAB                             ; 0336
```

```
                                                                K 10
OPC$TIMESTAMP                                            16-Sep-1984 01:57:57    VAX-11 Bliss-32 V4.0-742              Page  10
V04-000                                                  14-Sep-1984 12:50:59    [OPCOM.SRC]TIMESTAMP.B32;1                 (2)

                        00000000G  00               01  FB 0013E          CALLS   #1, SYS$FLUSH                       ;
                                           0000'    CF  D5 00145 15$:     TSTL    PURGE_LIMIT                         ; 0343
                                                    41  12 00149          BNEQ    18$                                 ;
                                                    7E  7C 0014B          CLRQ    -(SP)                               ; 0348
                                                    7E  D4 0014D          CLRL    -(SP)                               ;
                                           0000'    CF  9F 0014F          PUSHAB  JPI_WSITEMS                         ;
                                                    7E  7C 00153          CLRQ    -(SP)                               ;
                                                    7E  D4 00155          CLRL    -(SP)                               ;
                        00000000G  00               07  FB 00157          CALLS   #7, SYS$GETJPI                      ;
                                   5B               50  D0 0015E          MOVL    R0, STATUS                          ;
                                   47               5B  E9 00161          BLBC    STATUS, 19$                         ;
                                   50      0000G    CF  D0 00164          MOVL    SYI_SWPOUTPGCNT, SWAPO              ; 0355
              51          0000'    CF      0000'    CF  C1 00169          ADDL3   GPGCNT, PPGCNT, R1                  ; 0356
                                   51               0A  C0 00171          ADDL2   #10, R1                             ;
                                   50               51  D1 00174          CMPL    R1, SWAPO                           ;
                                                    03  18 00177          BGEQ    16$                                 ;
                                   51               50  D0 00179          MOVL    SWAPO, R1                           ;
                                   50               03  C4 0017C 16$:     MULL2   #3, SWAPO                           ; 0357
                                   51               50  D1 0017F          CMPL    R0, LIMIT                           ; 0358
                                                    03  15 00182          BLEQ    17$                                 ;
                                   50               51  D0 00184          MOVL    LIMIT, R0                           ;
                          0000'    CF               50  D0 00187 17$:     MOVL    R0, PURGE_LIMIT                     ;
                          0000G    CF         60    8F  8A 0018C 18$:     BICB2   #96, GLOBAL_STATUS                  ; 0364
                                                    7E  D4 00192          CLRL    -(SP)                               ; 0365
                                           FE68     CF  9F 00194          PUSHAB  TIME_STAMP                          ;
                                           0000G    CF  9F 00198          PUSHAB  WAIT_DELTA                          ;
                                                    04  DD 0019C          PUSHL   #4                                  ;
                        00000000G  00               04  FB 0019E          CALLS   #4, SYS$SETIMR                      ;
                                   5B               50  D0 001A5          MOVL    R0, STATUS                          ;
                                   09               5B  E8 001A8          BLBS    STATUS, 20$                         ;
                                                    5B  DD 001AB 19$:     PUSHL   STATUS                              ; 0367
                        00000000G  00               01  FB 001AD          CALLS   #1, LIB$STOP                        ;
                                                        04 001B4 20$:     RET                                         ; 0369

; Routine Size:  437 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 GLOBAL ROUTINE IMPLICITLY_CANCELED (RQST) =
1
1 !++
1 ! Functional description:
1 !
1 !       Check a given request to see if it has been implicitly canceled.
1 !       An implicit cancelation is defined as the requestor deleting the
1 !       reply mailbox without first having sent an explicit request cancelation
1 !       message to OPCOM.
1 !
1 ! Input:
1 !
1 !       RQST    : address of a request control block
1 !
1 ! Implicit Input:
1 !
1 !       None.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       None.
1 !
1 ! Side effects:
1 !
1 !       If the request has been implicitly canceled, it will be inserted
1 !       into a queue of canceled requests.  The queue will be processed later.
1 !
1 ! Routine value:
1 !
1 !       TRUE    : the request has been implicitly canceled
1 !       FALSE   : the request is still active
1 !--
1
2 BEGIN                                         ! Start of IMPLICITLY_CANCELED
2
2 MAP
2       RQST            : $ref_bblock;          ! Request control block
2
2 EXTERNAL ROUTINE
2       CLUSUTIL_SYSTEMID_EQUAL : JSB_R0R1;
2
2 EXTERNAL
2       GLOBAL_STATUS   : BITVECTOR [32],
2       CANCELED_RQST_Q : VECTOR,               ! List head of canceled requests
2       LCL_NOD         : $ref_bblock,
2       MBX_FAO         : $bblock;              ! FAO control string
2
2 LOCAL
2       MBX_NAME        : $bblock [MAX_DEV_NAM],! Mailbox device name buffer
2       MBX_DESC        : $desc_block,          ! Mailbox device name descriptor
2       DEV_CHAR        : $bblock [DIB$K_LENGTH],! Mailbox dev. char. buffer
2       CHAR_DESC       : $desc_block;          ! Mailbox dev. char. descriptor
2
```

```
;  429          0427   2  !
;  430          0428   2  ! Do not implicitly cancel requests from other nodes
;  431          0429   2  !
;  432          0430   2  IF .GLOBAL_STATUS [GBLSTS_K_IN_VAXcluster]
;  433          0431   2  THEN
;  434          0432   2      IF NOT CLUSUTIL_SYSTEMID_EQUAL (RQST [RQCB_T_SYSTEMID], LCL_NOD [NOD_T_NODE_SYSTEMID])
;  435          0433   2      THEN
;  436          0434   2          RETURN FALSE;                               ! Not disabled
;  437          0435   2  !
;  438          0436   2  !
;  439          0437   2  ! Check to see if the request has been implicitly canceled.
;  440          0438   2  ! The simplest way to do this is to attempt to get the device
;  441          0439   2  ! characteristics.  If the device no longer exists, then assume
;  442          0440   2  ! the user is no longer interested in the request.  First format
;  443          0441   2  ! the mailbox name from the information in the RQCB.
;  444          0442   2  !
;  445          0443   2  !
;  446          0444   2  MBX_DESC [DSC$W_LENGTH]  = MAX_DEV_NAM; ! Create a descriptor
;  447          0445   2  MBX_DESC [DSC$B_DTYPE]   = 0;
;  448          0446   2  MBX_DESC [DSC$B_CLASS]   = 0;
;  449          0447   2  MBX_DESC [DSC$A_POINTER] = MBX_NAME;
;  450          0448   2  $FAO (MBX_FAO, MBX_DESC, MBX_DESC, .RQST [RQCB_W_REPLYMBX]);
;  451          0449   2  CHAR_DESC [DSC$W_LENGTH]  = DIB$K_LENGTH;       ! Create a descriptor
;  452          0450   2  CHAR_DESC [DSC$B_DTYPE]   = 0;
;  453          0451   2  CHAR_DESC [DSC$B_CLASS]   = 0;
;  454          0452   2  CHAR_DESC [DSC$A_POINTER] = DEV_CHAR;
;  455          0453   3  IF ($GETDEV (DEVNAM=MBX_DESC, PRIBUF=CHAR_DESC))
;  456          0454   2  THEN
;  457          0455   2      !
;  458          0456   2      ! The reply mailbox still exists.
;  459          0457   2      !
;  460          0458   2      RETURN FALSE
;  461          0459   2  ELSE
;  462          0460   3      BEGIN
;  463          0461   3      !
;  464          0462   3      ! The reply mailbox no longer exists.  Assume request canceled.
;  465          0463   3      !
;  466          0464   3      RQST [RQSTS_V_IMPCANCEL] = TRUE;
;  467          0465   3      INSQUE (RQST [RQCB_L_DSBLFLINK], CANCELED_RQST_Q);
;  468          0466   3      RETURN TRUE;
;  469          0467   2      END;
;  470          0468   2
;  471          0469   1  END;                                            ! End of IMPLICITLY_CANCELED


                                                                 .EXTRN  CLUSUTIL_SYSTEMID_EQUAL
                                                                 .EXTRN  CANCELED_RQST_Q
                                                                 .EXTRN  LCL_NOD, MBX_FAO
                                                                 .EXTRN  SYS$FAO, SYS$GETDEV

                                           0004 00000            .ENTRY  IMPLICITLY_CANCELED, Save R2                     ; 0370
                    5E       FF3C    CE    9E 00002              MOVAB   -196(SP), SP                                     ;
                    15       0000G   CF    E9 00007              BLBC    GLOBAL_STATUS+1, 1$                              ; 0430
      51     0000G  CF 00000050      8F    C1 0000C              ADDL3   #80, LCL_NOD, R1                                 ; 0432
      50        04  AC                1C   C1 00016              ADDL3   #28, RQST, R0                                    ;
                                    0000G  30 0001B              BSBW    CLUSUTIL_SYSTEMID_EQUAL                          ;
```

```
OPC$TIMESTAMP                                          N 10
V04-000                                   16-Sep-1984 01:57:57   VAX-11 Bliss-32 V4.0-742              Page  13
                                          14-Sep-1984 12:50:59   [OPCOM.SRC]TIMESTAMP.B32;1                  (3)

                      4F           50  E9 0001E         BLBC    R0, 2$                                   ;
               7C     AE     40    8F  9A 00021 1$:     MOVZBL  #64, MBX_DESC                            ; 0444
               BC     AD     C0    AD  9E 00026         MOVAB   MBX_NAME, MBX_DESC+4                     ; 0447
                      52     04    AC  D0 0002B         MOVL    RQST, R2                                 ; 0448
                      7E     2E    A2  3C 0002F         MOVZWL  46(R2), -(SP)                            ;
                             B8    AD  9F 00033         PUSHAB  MBX_DESC                                 ;
                             B8    AD  9F 00036         PUSHAB  MBX_DESC                                 ;
                          0000G    CF  9F 00039         PUSHAB  MBX_FAO                                  ;
          00000000G   00           04  FB 0003D         CALLS   #4, SYS$FAO                              ;
                      6E     74    8F  9A 00044         MOVZBL  #116, CHAR_DESC                          ; 0449
               04     AE     08    AE  9E 00048         MOVAB   DEV_CHAR, CHAR_DESC+4                    ; 0452
                                   7E  7C 0004D         CLRQ    -(SP)                                    ; 0453
                             08    AE  9F 0004F         PUSHAB  CHAR_DESC                                ;
                                   7E  D4 00052         CLRL    -(SP)                                    ;
                             B8    AD  9F 00054         PUSHAB  MBX_DESC                                 ;
          00000000G   00           05  FB 00057         CALLS   #5, SYS$GETDEV                           ;
                      0F           50  E8 0005E         BLBS    R0, 2$                                   ;
               2A     A2           01  88 00061         BISB2   #1, 42(R2)                               ; 0464
            0000G     CF   008C    C2  0E 00065         INSQUE  140(R2), CANCELED_RQST_Q                 ; 0465
                      50           01  D0 0006C         MOVL    #1, R0                                   ; 0466
                                       04 0006F         RET                                              ; 0460
                                   50  D4 00070 2$:     CLRL    R0                                       ; 0469
                                       04 00072         RET                                              ;

; Routine Size:  115 bytes,    Routine Base:  $CODE$ + 01B5
```

```
1 GLOBAL ROUTINE IMPLIED_CANCEL : NOVALUE =
1
1 !++
1 ! Functional description:
1 !
1 !       For all requests on the canceled request queue, create a
1 !       cancellation message from the information in the request
1 !       control block, and CALL the request cancellation handler
1 !       as if the user had sent the cancellation message.
1 !
1 ! Input:
1 !
1 !       None.
1 !
1 ! Implicit Input:
1 !
1 !       CANCELED_RQST_Q : The list head of all implicitly canceled requests.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       None.
1 !
1 ! Side effects:
1 !
1 !       All interested operators will be notified of the canceled requests.
1 !       As this is done, implicitly disabled operators may be discovered.
1 !       Those operators will be placed on the implicit disable queue and
1 !       be processed later.
1 !
1 ! Routine value:
1 !
1 !       None.
1 !--
1
2 BEGIN                                                 ! Start of IMPLIED_CANCEL
2
2 EXTERNAL ROUTINE
2       CNCL_HANDLER    : NOVALUE,                      ! Old CANCEL message handler
2       NOTIFY_LISTED_OPERATORS;                        ! Notify a list of operators
2
2 EXTERNAL
2       CANCELED_RQST_Q : VECTOR;                       ! List head of canceled requests
2
2 LITERAL
2       MSG_HDR_SIZE    = ($BYTEOFFSET(RQCB_B_RQSTCODE) - $BYTEOFFSET(RQCB_W_MSGTYPE)),
2       OLD_MSG_SIZE    = 8,
2       MSG_BUF_SIZE    = MSG_HDR_SIZE + OLD_MSG_SIZE;
2
2 MACRO
2       REQUEST_TYPE    = MSG_HDR_SIZE,   0,  8, 0%,
2       TARGET_MASK     = MSG_HDR_SIZE+1, 0, 24, 0%,
2       REQUEST_ID      = MSG_HDR_SIZE+4, 0, 32, 0%;
2
```

```
;  530          0527  2 LOCAL
;  531          0528  2         CANCEL_MSG_BUF  : $bblock [MSG_BUF_SIZE],! CANCEL request message buffer
;  532          0529  2         CANCEL_MSG_DESC : $desc_block,           ! CANCEL request descriptor
;  533          0530  2         RQST            : $ref_bblock;           ! Request control block
;  534          0531  2
;  535          0532  2 !
;  536          0533  2 !
;  537          0534  2 ! Create the message buffer descriptor.  We need do this only once.
;  538          0535  2 !
;  539          0536  2 !
;  540          0537  2 CANCEL_MSG_DESC [DSC$W_LENGTH]  = MSG_BUF_SIZE;
;  541          0538  2 CANCEL_MSG_DESC [DSC$B_DTYPE]   = 0;
;  542          0539  2 CANCEL_MSG_DESC [DSC$B_CLASS]   = 0;
;  543          0540  2 CANCEL_MSG_DESC [DSC$A_POINTER] = CANCEL_MSG_BUF;
;  544          0541  2 !
;  545          0542  2 !
;  546          0543  2 ! For all requests on the queue, create a cancel message
;  547          0544  2 ! (old format) and call the cancel request handler.
;  548          0545  2 !
;  549          0546  2 !
;  550          0547  2 WHILE NOT REMQUE (.CANCELED_RQST_Q, RQST) DO
;  551          0548  3    BEGIN
;  552          0549  3    RQST = .RQST - ($BYTEOFFSET(RQCB_L_DSBLFLINK) - $BYTEOFFSET(RQCB_L_FLINK));
;  553          0550  3    CH$MOVE (MSG_HDR_SIZE, RQST [RQCB_Q_MSGTYPE], CANCEL_MSG_BUF);
;  554          0551  3    CANCEL_MSG_BUF [REQUEST_TYPE] = OPC$_RQ_CANCEL;
;  555          0552  3    CANCEL_MSG_BUF [TARGET_MASK]  = .RQST [RQCB_L_ATTNMASK1];
;  556          0553  3    CANCEL_MSG_BUF [REQUEST_ID]   = .RQST [RQCB_L_RQSTID];
;  557          0554  3    CNCL_HANDLER (CANCEL_MSG_DESC);
;  558          0555  2    END;
;  559          0556  2
;  560          0557  1 END;                                                 ! End of IMPLIED_CANCEL


                                                                          .EXTRN  CNCL_HANDLER

                                                007C 00000                .ENTRY  IMPLIED_CANCEL, Save R2,R3,R4,R5,R6            ; 0470
                         5E                 34   C2 00002                SUBL2   #52, SP                                        ;
                                            2E   DD 00005                PUSHL   #46                                            ; 0537
                   04    AE         08      AE   9E 00007                MOVAB   CANCEL_MSG_BUF, CANCEL_MSG_DESC+4              ; 0540
                         56      0000G      DF   0F 0000C 1$:            REMQUE  @CANCELED_RQST_Q, RQST                         ; 0547
                                            24   1D 00011                BVS     2$                                             ;
                         56       FF74      C6   9E 00013                MOVAB   -140(R6), RQST                                 ; 0549
             08    AE          2C      A6   26   28 00018                MOVC3   #38, 44(RQST), CANCEL_MSG_BUF                  ; 0550
                               2E      AE   05   90 0001E                MOVB    #5, CANCEL_MSG_BUF+38                          ; 0551
   2F    AE            18      00      5C   A6   F0 00022                INSV    92(RQST), #0, #24, CANCEL_MSG_BUF+39           ; 0552
                         32    AE      64   A6   D0 00029                MOVL    100(RQST), CANCEL_MSG_BUF+42                   ; 0553
                                            5E   DD 0002E                PUSHL   SP                                             ; 0554
                      0000G    CF           01   FB 00030                CALLS   #1, CNCL_HANDLER                               ;
                                            D5   11 00035                BRB     1$                                             ; 0547
                                                 04 00037 2$:            RET                                                    ; 0557

; Routine Size:  56 bytes,    Routine Base:  $CODE$ + 0228
```

```
1 GLOBAL ROUTINE IMPLIED_DISABLE : NOVALUE =
1
1 !++
1 ! Functional description:
1 !
1 !       For all implicitly disabled operators create an operator disable
1 !       message using the info in the operator control block, and CALL the
1 !       operator enable message handler as if the user had sent the message.
1 !       Note that notification of the operator disable in NOT sent to the
1 !       operator.  This is because the terminal is no longer an operator
1 !       terminal, and the user now at the terminal doesn't need to see the
1 !       message.
1 !
1 ! Input:
1 !
1 !       None.
1 !
1 ! Implicit Input:
1 !
1 !       DISABLED_OPER_Q : The list head of all implicitly disabled operators.
1 !
1 ! Output:
1 !
1 !       None.
1 !
1 ! Implict output:
1 !
1 !       None.
1 !
1 ! Side effects:
1 !
1 !       As operators are disabled, more implicitly disabled operators may
1 !       be discovered.  If so, they will be inserted on the queue, and
1 !       processed in turn.  Likewise, as operators are disabled, some requests
1 !       may lose operator coverage.  These requests will be canceled and
1 !       the user notified.
1 !
1 ! Routine value:
1 !
1 !       None.
1 !--
1
2 BEGIN                                           ! Start of IMPLIED_DISABLE
2
2 EXTERNAL
2       DISABLED_OPER_Q : VECTOR;               ! List head of disabled operators
2
2 LOCAL
2       STATUS;
2
2 STATUS = 1;                                     ! *** TEMP ***
2
1 END;                                            ! End of IMPLIED_DISABLE


                                                  .EXTRN  DISABLED_OPER_Q
```

```
                                                     0000 00000          .ENTRY  IMPLIED DISABLE, Save nothing               ; 0558
                                       50         01  D0 00002          MOVL    #1, STATUS                                   ; 0608
                                                      04 00005          RET                                                  ; 0610

; Routine Size:  6 bytes,    Routine Base:  $CODE$ + 0260


;  615          0611  1
;  616          0612  1 END                                       ! End of TIMESTAMP
;  617          0613  0 ELUDOM




;                                PSECT SUMMARY
;
;       Name                       Bytes                      Attributes
;
;  $GLOBAL$                            4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                              40  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                            614  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)




;                          Library Statistics
;
;                                              -------- Symbols --------        Pages        Processing
;       File                                   Total    Loaded   Percent        Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1               18619        21         0        1000           00:01.8
;  _$255$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1         633        31         4          43           00:00.9




;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:TIMESTAMP/OBJ=OBJ$:TIMESTAMP MSRC$:TIMESTAMP/UPDATE=(ENH$:TIMESTAMP)

;  618          0614  0
; Size:         614 code + 44 data bytes
; Run Time:        00:15.3
; Elapsed Time:    00:55.5
; Lines/CPU Min:    2406
; Lexemes/CPU-Min: 16295
; Memory Used:  155 pages
; Compilation Complete
```

0292 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PASDEF
LIS

PASI02
LIS

STATUS
LIS

UNKNOWN
LIS

SHUTDOWN
LIS

SHARE
LIS

PASI01
LIS

PASI03
LIS

PASCAL

PASCLINE
LIS

TIMESTAMP
LIS